THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 61, No. 6, July-August 1982
Printed in U.S.A.

Automated Repair Service Bureau:

Second-Generation Mechanized Loop Testing System—A Distributed Microprocessor Application

By H. RUBIN

(Manuscript received December 2, 1980)

The Mechanized Loop Testing (MLT) system is that part of the Automated Repair Service Bureau that provides automatic acquisition and analysis of electrical test data for customer telephone loops. The MLT-2 system is a second-generation MLT that has as its basic architectural features communication, loop access, and loop test distributed as closely as possible to the point of testing. Architectural components include a wire center-based Loop Testing System (LTS) and a centrally located Data Communication Network (DCN). Each LTS contains communication, access, and test capabilities, and is logically connected by the DCN to each controlling minicomputer (up to 12). The LTS and DCN are each composed of multiple microprocessor-based circuits. The architecture of MLT-2 is presented. Particular attention is given to the subjects of partitioning both hardware and software, to the development of change-tolerant software, and to intrasystem communication capabilities powerful enough to support a large number of distributed processors. In addition, special measurement techniques employed by MLT-2 that take advantage of analog and digital large-scale integration technology are discussed. Operational scenarios are included for an appreciation of how the MLT-2 system works.

I. INTRODUCTION

At Bell Telephone Laboratories, Mechanized Loop Testing (MLT) is a generic term used to describe that part of the Automated Repair

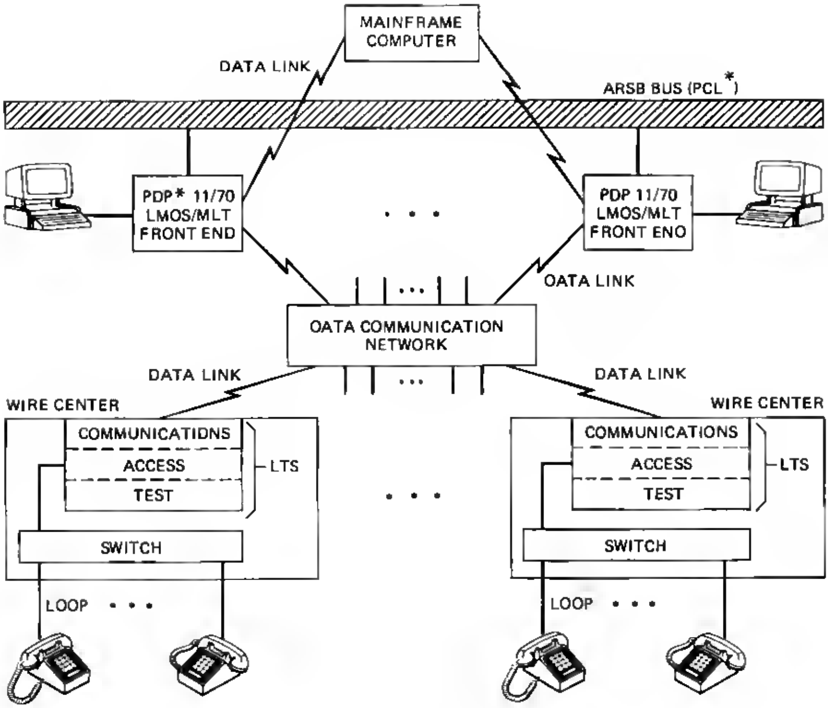
Service Bureau (ARSB) that provides automatic acquisition and analysis of electrical test data for customer telephone loops. As pointed out in Refs. 1 and 2, the acquired data are translated into an equivalent electrical circuit model, and matched against the Loop Maintenance Operations System (LMOS) data base information in an attempt to isolate loop faults or to determine that the loop is operating properly. The first such system, MLT-1, is described in Ref. 2. The MLT-2 system is an alternative solution to the loop testing problem which takes advantage of recent technological advances, and can be used either to augment an existing MLT-1 installation or to provide the total loop testing function in a given loop environment.

There are two major reasons for the development of an alternative MLT system. First, the Loop Testing Frame[2] (LTF) of MLT-1 does not represent a cost effective solution in very low population density areas.[3] Second, Bell Operating Companies (BOCs) have recently started consolidating their testing bureaus; this requires the relocation of the associated manual testboard positions (Local Test Desk No. 14 and No. 16). Relocation of the electromechanical testboards is an expensive operation because they were not designed to be moved from their initial installation site. For reasons explained in Ref. 2, MLT-1 does not eliminate completely the manual testboard system. Consequently, it has become important that the new automated testing system replace the manual system. The MLT-2 system is cost effective in a small wire center environment, sophisticated enough to eliminate the existing manual systems, and it makes loop testing independent of how the testing bureaus are organized within the BOC. Distributed processing techniques using microprocessor-based circuitry allow the realization of these system characteristics.

Automated loop testing that is coupled to LMOS is a rather natural application of distributed computing; subscriber loops are dispersed over a wide geographical area, whereas the LMOS data-base is centralized within the front-end (FE) processors. For economic reasons, the MLT-1 testing vehicle [the LTF[2]] is designed to take advantage of functional concentration to realize economies of scale. Hence, distributed processing in MLT-1 stops at the MLT-1 Controller.[2] The MLT-2 system relies on the use of microprocessors and new techniques for loop measurements to meet the challenges of cost and performance, and to take more complete advantage of the distributed nature of the testing problem.

II. AN OVERVIEW OF THE MLT-2 ARCHITECTURE

To test telephone loops (as opposed to building a complete testing system with its human interface),[2] three basic functions are required: access, test, and communications. These three basic functions are



* REGISTERED TRADEMARK OF DIGITAL EQUIPMENT CORPORATION

Fig. 1—Architecture of MLT-2.

present in all manual and automatic testing systems. The testing system has to gain control of the loop and bave physical access to it in order to test it. In addition, the testing system has to have two-way communications with a central controller that determines the testing pattern and collects the test results for analysis. The reader can recognize these basic functions in the MLT-1 system described in Ref. 2. The main architectural technique used in the design of MLT-2 is to distribute the access, test, and communications functions as closely as possible to the point of testing, i.e., to the loop itself. The use of this technique tends to minimize the data flow in the system, and is consistent with design strategies for functional distribution that have been advocated elsewhere.[4,5]

Figure 1 shows a block diagram of the MLT-2 architecture. The ARSB high-performance Parallel Communication Link (PCL)[6] and connections to the mainframe computer are shown for completeness. (See Refs. 7 and 8 for a discussion of the PCL and the mainframe computer.) In Fig. 1, each wire center served by the system contains a microprocessor-based Loop Testing System (LTS) that consists of access, test,

and communications capabilities. The use of microprocessor technology makes it economically possible to distribute these functions and their control on a wire center basis, where the loops terminate on the central office equipment. The communications function that resides in each LTS is incomplete without the Data Communication Network (DCN), which is part of the architecture in Fig. 1. The DCN allows any one of the PDP* 11/70 LMOS/MLT computers (FEs) to communicate with the LTS in any wire center served by the system. The DCN is itself a microprocessor-based distributed processing machine that off-loads communications processing for all FEs (up to twelve) attached to the PCL. The MLT-1 architecture restricts each FE to serve a unique subset of customer loops, whereas the architecture of MLT-2 allows any FE to test any customer loop.

The MLT-2 system operates in the following manner. Repair Service Bureau personnel have a CRT interface to the LMOS/MLT system, and are able to input data (a telephone number to be tested) and receive output (test results) from the system.[9] Each CRT is connected by data link to one of the FEs in the ARSB. If the user requests MLT testing, tbe MLT software that resides on the FE interacts with LMOS functions on that machine to retrieve the data base information for the loop to be tested. This information is passed to an application process tbat initiates and guides the loop access and testing. The application process may contain the adaptive loop testing algorithm discussed in Ref. 1, or it may contain software to implement interactive test control and other functions that allow the elimination of the manual test board systems referred to earlier.

Because of the intelligence located in the LTS microprocessor circuits, only high-level commands need to be generated by the FE software. The first command requests tbe LTS to access a specified telephone number. The message header contains a parameter that identifies the LTS data link for the system to use. This message is routed by the DCN to the appropriate LTS data link (see Fig. 1). When access is completed, the response is routed througb the DCN to the FE that requested the access. Subsequent message transactions that occur between the LTS and the FE involve high-level requests for tests to be performed, followed by detailed responses containing raw test data (the amount of current tbat was measured on the loop wires when a particular source was applied to the loop, etc.). See Ref. 2 for a description of the tests typically performed by MLT. The last request made by the FE is to have the LTS drop the access to the loop under test. The number of loops that may be accessed simultaneously at any LTS site and the

* Registered trademark of Digital Equipment Corporation.

number of simultaneous tests that may be in progress at a given LTS are discussed in the sequel.

The MLT-2 architecture has the following attributes:

(*i*) Since FEs can control testing on any loop, they provide active backup to each otber, thereby increasing system reliability.

(*ii*) High-level commands are passed from the FE to the LTS, thereby minimizing the volume of communications required.

(*iii*) The FE MLT function controls the testing of a loop, but does not control the details of each test. Less processing overhead means that higher system throughput can be achieved.

(*iv*) Increased throughput means that many more wire centers can be handled by the MLT-2 system than by the MLT-1 system. In addition, MLT control software and LMOS software can share the same FE.

(*v*) Distribution of function to the point of testing eliminates long test trunk connections.

This overview shows the basic design philosophy of MLT-2, and illustrates the benefits that arise from adopting a distributed processing approach to loop testing. However, many problems need to be overcome in implementing the architecture. The distributed functions have to be made cost effective in small loop environments, and the communications and control mechanisms must be sufficient to support the interactions of many distributed intelligences. Hardware and software functions need to be partitioned so as to reduce module complexity in order to simplify module design and maintenance. The following sections describe the microprocessor-based LTS and DCN in detail from both a hardware and software perspective, and it is shown how the "divide and conquer" technique is used to render a practical design.

III. MLT-2 HARDWARE IMPLEMENTATION

3.1 Loop Testing System

The wire center-based LTS is a collection of loosely coupled* distributed microprocessors organized to perform tbe communications, loop access, and loop testing functions. The block diagram in Fig. 2 shows an LTS controller that is responsible for communications witb the DCN, for control of circuits used to provide a talk function for ARSB personnel, and for local control of the other LTS processors. The port controller is responsible for the access function. It provides tip, ring, and sleeve lead control for connections to no-test trunk circuits that enable MLT to interface to the switching machine. The Precision Measurement Unit (PMU) is a general-purpose testing instrument that is used to

* The term "loosely coupled" is used here to denote an organization of processors that share no common memory but communicate by passing messages over a serial or parallel interface.

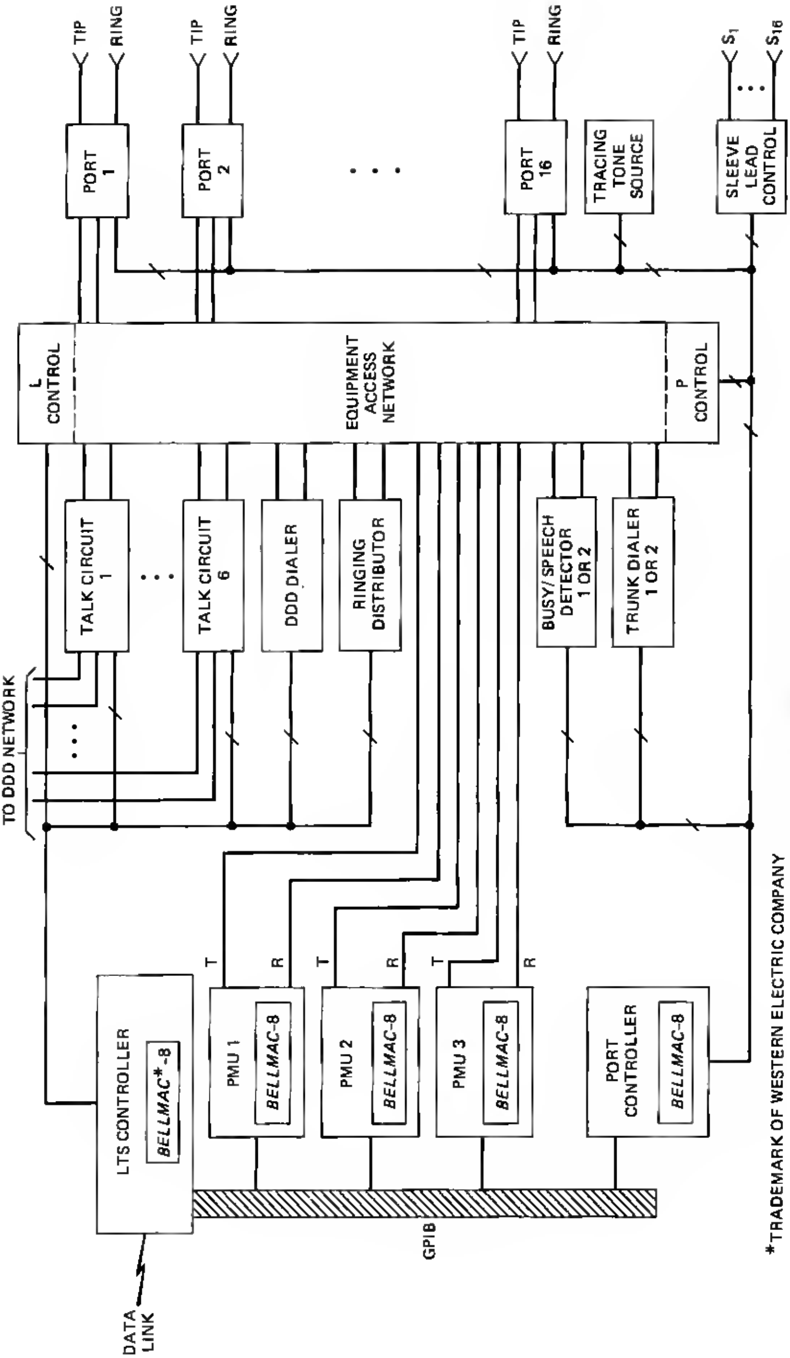


Fig. 2—Loop Testing System.

make loop measurements. Each LTS may contain from one to three PMUs. Each PMU is based on a *BELLMAC*†-8 microprocessor[10] module, as are the LTS and port controllers. All processors in the LTS are loosely coupled, and use an IEEE STD-488 General Purpose Interface Bus (GPIB)[11] as the interprocessor communication mechanism. The GPIB is controlled by the LTS controller. The data link connection to the DCN is a synchronous, full duplex 1200- or 2400-baud link utilizing the BX.25 level 2 communication protocol.[12]

The system is designed to be modular so that wire centers ranging from roughly 1000 to 100,000 lines can be served economically. As the wire center size increases, more PMUs can be added (up to three per LTS), up to sixteen port circuits (interfaces to no-test trunk circuits) can be accommodated, and the Equipment Access Network (EAN) internal to the LTS can be expanded to enable any of the PMUs, dialers, talk circuits, etc., to be connected to any port circuit. Hence, the largest size LTS can have up to sixteen loops simultaneously accessed for testing, and can time share the three identical PMUs to perform requested tests. The largest size LTS, therefore, contains five *BELLMAC*-8 microprocessor modules, as well as several single-chip microcomputers (four per PMU) to perform special functions for the PMU.

The separation of the testing function from the access and communications functions simplifies the LTS organizational structure in the case where multiple PMUs are required to handle a given testing traffic load. The assignment of access and communications to separate processors increases significantly the throughput of the LTS. With the arrangement shown in Fig. 2, communications, access, and testing can be going on simultaneously. It is also evident that distributing the local processing provides additional memory space for the LTS functions. A system such as MLT has great potential for functional growth, and additional program memory size is an asset.

3.1.1 Loop Testing System operation

The reader is referred to Fig. 2 for the following discussion. The LTS controller implements the BX.25 level 2 protocol function on the data link. Received messages are parsed and interpreted by the LTS controller. An access request causes the LTS controller to establish some data in RAM that is used to track and time the request. A message is then generated, and passed over the GPIB interface to the port controller. The port controller proceeds to access the loop specified in the message by attaching a trunk dialer to an appropriate port, dialing the telephone number, and attaching a busy/speech detector circuit to determine whether the loop is idle. When loop access is obtained, the

† Trademark of Western Electric Company.

port controller sends a response across the GPIB, and the LTS controller proceeds to satisfy any test requests that may have been contained in the original request message. If no test requests are present, a response is generated for the FE, and is transmitted over the BX.25 link. At this point, the loop is accessed, and the LTS controller awaits subsequent requests for tests.

Most test requests require the services of a PMU. However, some requests can he satisfied by either the LTS controller or the port controller and their associated circuitry. Test requests are coded so that the LTS controller can determine which LTS circuits can satisfy the request. The LTS controller, therefore, acts as a resource manager for the LTS.

Fig. 2 indicates that the port controller can perform sleeve lead manipulation, can apply a tone source to the customer loop, and (not shown) can monitor the loop in a coarse sense for a shorted or open condition on tip and ring leads. Sleeve lead control is used to signal the trunk circuit in order to pull in the customer line circuit for testing or to gain access to certain types of testing circuits associated with the central office. Tone is applied to the loop to help outside plant personnel locate a particular wire pair in a cable. Coarse detection of shorts and opens on a customer loop is also required as an aid in pair identification by outside plant personnel. Although the latter two features can be implemented with a PMU, the duration of the tone application or monitor function is in the order of minutes. Consequently, MLT-2 does not attempt to use the sophisticated PMU for these purposes. The port controller functions mentioned above allow MLT-2 to replace the manual testboard system.

Another feature required to allow replacement of the manual system is the ability to alternately talk to a customer (or craft person) and test the customer loop. This feature is provided by the LTS controller via the LTS talk circuits. If an interactive session is desired, the initial access request contains, in addition to the telephone number of the loop to be tested, the numher of a telephone that is associated with the craft person at the ARSB work center. The LTS passes the customer telephone numher over the GPIB to the port controller as before, but now proceeds to use a dialer to place a call over the DDD network to the work center. When both connections are made, the LTS can be commanded to ring the customer loop, and, subsequent to ring-trip detection, connect the test trunk and the DDD path through one of the LTS talk circuits. The craft person can enter requests through the CRT, and cause the LTS to break the talk path temporarily, perform the requested test, and reconnect the talk path.

Tests and features other than those mentioned above require the use of a PMU.

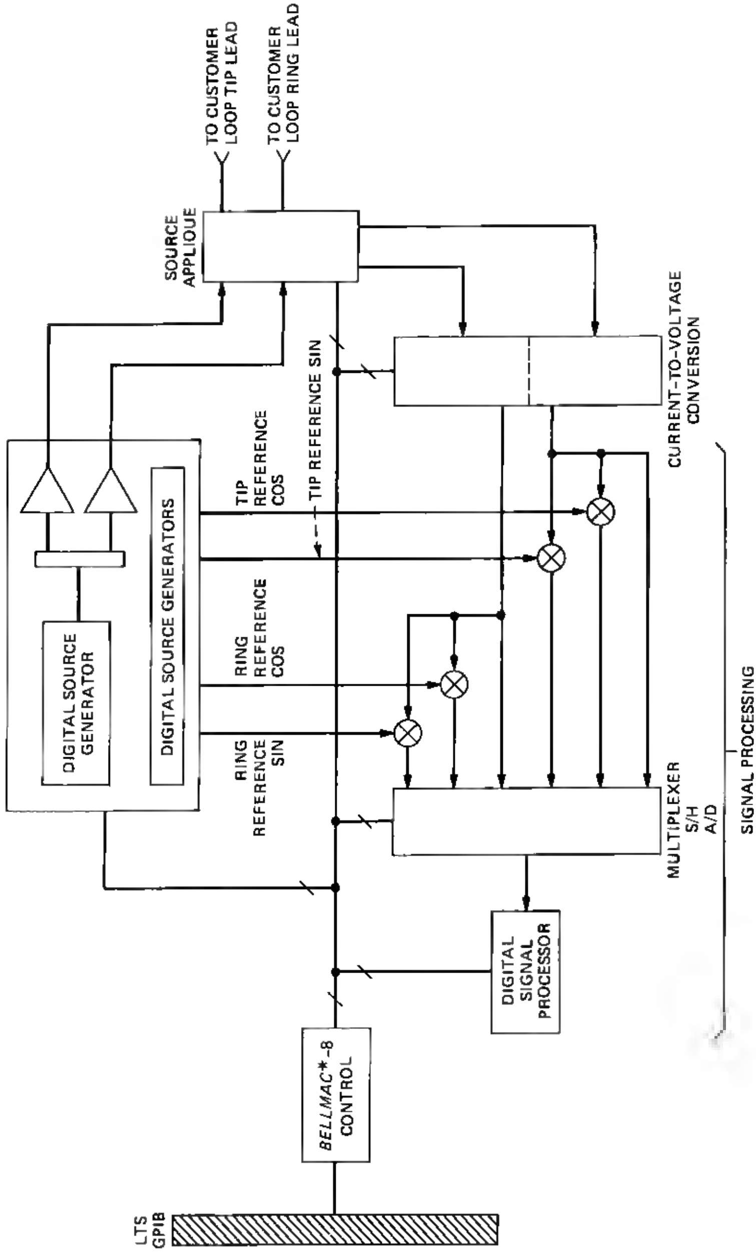
3.1.2 Precision Measurement Unit

No part of the PMU is dedicated to performing any particular test. Fig. 3 presents a block diagram of the PMU circuitry, and shows the *BELLMAC*-8 controller, a source generation section, a source applique (terminations), loop current detectors, and a signal processing section. The PMU is capable of performing all tests that MLT-1 can perform with its several test circuit types,[2] and can also perform many tests that MLT-1 cannot perform. The *BELLMAC*-8 processor receives test requests via the GPIB interface, sets up the test by interfacing to the components shown in Fig. 3, and transmits the results across the GPIB when the test is completed.

The source generation section of the PMU consists of a set of three single-chip microcomputers. Each microcomputer can generate digital samples of quadrature sinewaves at programmed frequencies ranging from 1 to 3200 Hz in 1-Hz steps. A table lookup algorithm is used for this purpose. The digital samples are converted to analog form by means of digital-to-analog converters, possibly combined with a dc level, and applied to a power amplifier. Signals up to a 135-volt peak and up to 125 mA, from dc to 3200 Hz, can be generated by this circuitry. The single-chip signal generators can produce several functions, including pulsed waveforms, swept waveforms, and dual frequency waveforms.

Mechanized loop testing systems perform mainly admittance measurements.[2] Hence, in order to test a loop, the signal voltage sources are applied to the tip and ring leads, and the resultant current flow is measured. In MLT-2 the current is measured by means of a magnetic technique that uses components that are substantially smaller and more efficient than those used in MLT-1 for the same purpose.[2] The outputs of the two magnetic sensing circuits are voltages proportional to the currents in the loop conductors. The two channels can be configured to produce the metallic and/or longitudinal loop currents.

The signal processing section of the PMU consists of signal conditioning circuitry for each current sensing channel, an analog multiplexer, an analog-to-digital converter, and a digital filter. Of the three microcomputers in the source generation section, only one is used to generate the voltage waveform that is applied to the loop. The other two are used to generate inputs to the two signal conditioning channels. Waveforms at precisely the frequency applied to the loop can be generated in quadrature and used to synchronously demodulate the voltage outputs from the current sensors. Analog multipliers are used for this purpose. The PMU circuit phase shifts can be accommodated simply by shifting the phases of the demodulator sources relative to the loop source. Quadrature waveforms are used to generate demodulated voltages that are proportional to the real and imaginary compo-



*TRADEMARK OF WESTERN ELECTRIC COMPANY

Fig. 3—Precision Measurement Unit.

nents of the loop current. Harmonics of the frequency applied to the loop can be generated and used to detect nonlinearities in the loop current. The use of two signal conditioning channels and two demodulating sources allows the PMU to make multiple measurements simultaneously. The reader should note that in this scheme, all results are dc values, either initially or after the demodulation and filtering process.

As the above paragraph suggests, the signal processing section's input circuitry can produce several outputs simultaneously, depending on the test being performed. An analog multiplexer is used to select the desired outputs, and feed tbem to a sample/hold and A/D converter circuit. Digital samples are fed to a Digital Signal Processor (DSP) chip,[13] where most of the filtering in tbe system is performed. The DSP contains several digital filter programs and a dynamic settling algorithm that can decide when a final value has been obtained from a measurement. Test results are passed from the DSP device to the PMU control processor, and are sent over tbe GPIB to the LTS controller.

The general-purpose design of the PMU is evident in the above discussion when one realizes that, within the voltage and frequency limits specified, the PMU can make measurements to characterize any three-terminal network. The PMU intelligence is also used to provide self-calibration functions and a sanity/diagnostic function.

The description of the LTS shows how the hardware is mapped onto the physical needs of the loop testing problem. The result is a loosely coupled distributed architecture in which each processor's operational details are hidden from the others. Getting something done in the system requires only that a message be passed between processors. The technique of mapping the hardware onto the problem is extended to the software as well, and represents a continuing theme in the MLT-2 design. The result is an easily understood and maintainable system.

3.2 Data Communication Network

Figure 1 indicates that the DCN has a very simple functional requirement, namely, to route messages between any FE and any LTS. The architecture to be described below allows from one to twelve FEs to exchange data with up to 768 LTSs, using the BX.25 level 2 protocol. (The limit of 768 is determined by physical design constraints and by expected needs of the application. The total capacity of the architecture to be presented is 1800 LTS data links.) Two other requirements include having the ability to operate when remoted from the ARSB FE complex, and having enough redundancy to withstand a single failure.

Figure 4 shows a three-tiered multiple processor architecture that realizes the DCN function. The microcomputers based on the *BELL*-

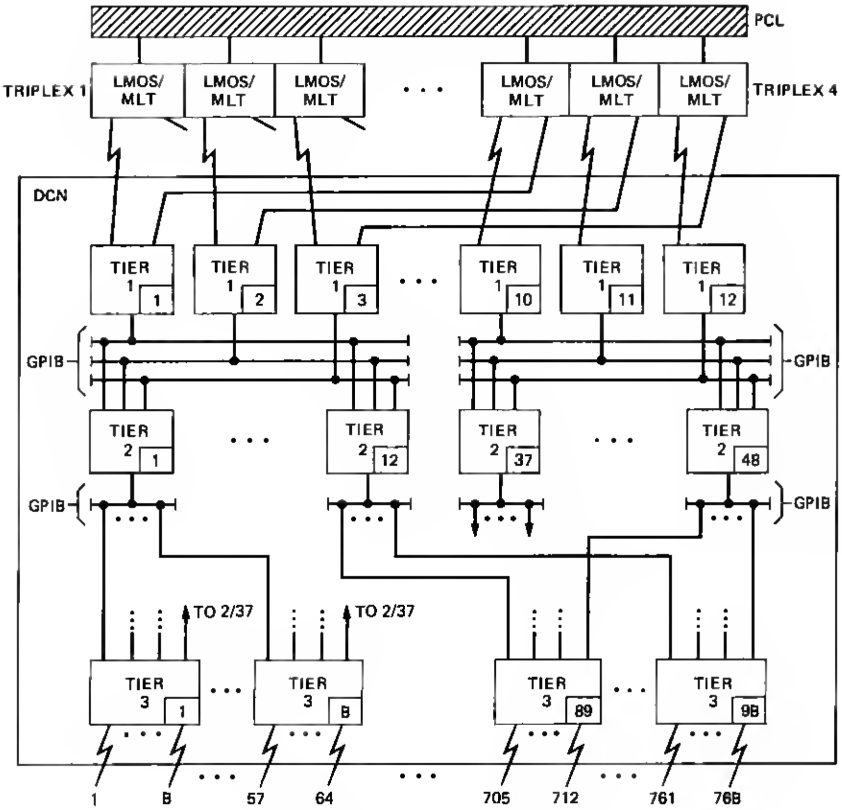


Fig. 4—Data Communication Network.

MAC-8 microprocessor are loosely coupled, and the design is modular in that eight LTS data links can be added at a time, and/or a single FE can be added to the system until its capacity is reached. Only five separate circuit pack codes are required to assemble the DCN.

Each LMOS/MLT FE is connected to two distinct DCN Tier 1 circuits via synchronous 9600-baud BX.25 data links. Dual active data links are used to meet reliability and throughput requirements. The use of data links for the FE interface allows the DCN to be remoted from the ARSB computer complex. Each third-tier circuit in the DCN provides eight 1200- or 2400-baud BX.25 data links that interface to the LTSs. The second-tier circuits serve to interface the first-tier and third-tier microcomputer circuits via IEEE STD-488 busses.

As can be seen from Fig. 4, the FEs are grouped in triplexes.* The

* The triplex configuration is derived from the LMOS application in which two FEs are active and one serves as a backup. All three share a common set of data links to user terminals.

second-tier DCN circuits are grouped in units of from one to twelve circuits and are dedicated to two distinct triplexes. Each second-tier function has a GPIB interface to each of three first-tier functions. The first-tier functions are backed up by virtue of the existence of two paths into the DCN for each FE. The second-tier functions are backed up only if more than one triplex is connected to the network, for in that case, an additional set of second-tier functions is used. In general, two unique paths exist in the DCN for message traffic between any FE and the eight LTSs served from a particular third-tier function. Message re-routing is used to deliver messages in the face of single circuit failures.

Figure 4 shows that the third-tier DCN circuits have GPIB interfaces to each set of second-tier functions present in the system (up to four GPIBs per third-tier function). Third-tier DCN functions are not backed up, and consequently, a single failure can result in the loss of communications up to eight LTSs. Sanity and diagnostic software functions are provided to identify these failures and help correct them within reasonable time limits.

Message routing in the DCN is accomplished by having two bytes in the message header contain an LTS data link identifier. This identifier is filled by the FE when it constructs the message. When a message passes into the DCN, the first-tier microcomputer software fills a third byte in the header that is reserved for the FE identifier. The response message from the LTS contains the same routing information as the original request message. Hence, the FE that generates the request receives the response.

The MLT-2 system can be seen to consist of a large number of microcomputer elements that are connected by communications facilities, including data links and local IEEE STD-488 busses. A typical MLT application may contain roughly three hundred *BELLMAC*-8 microprocessors; the largest installations may contain over a thousand *BELLMAC*-8 microprocessors. Functional decomposition is used to render the hardware design of such a system comprehensible. Similar techniques are applied to the software designs for these microprocessor subsystems.

IV. THE MLT-2 MICROCOMPUTER SOFTWARE

When the PMU receives a request to perform a test, it proceeds to do that function and that function only. Its operation is seen to be "single-threaded" in that it completes its activity without interruption. By contrast, all other MLT-2 microprocessor environments are "multi-threaded." In the DCN circuits, I/O operations can be occurring simultaneously over several different interfaces. In the LTS controller and port controller, activities for up to sixteen loop accesses may be in

different states of completion at the same instant. Furthermore, the operations required for interfacing to the central office equipment or for controlling the data link have many spans of time during which activity ceases before the *BELLMAC*-8 microprocessor needs to perform the next step. Hence, the MLT-2 microcomputer circuits operate for the most part in a multitasking environment. To cope with this environment, a small operating system (designated m8os) has been developed for MLT-2, and provides the multitasking facility, intertask communications via messages and semaphores, and buffer management. A user-defined interrupt structure is also supported. An attempt has been made to keep m8os as small and as fast as possible. The size of m8os is just 2200 bytes of text and data.

The provision of a multitasking environment facilitates the partitioning of the software into entities known as "tasks." Each user task is known to the operating system, and can be scheduled to execute when there is some operation to be performed by that task. Usually, a semaphore is set or a message is passed by some task or I/O that alerts m8os to the need to schedule a particular task. The main advantage to partitioning the software is to create functions of manageable size. Another important advantage is to create a structure that is change tolerant; when something in the environment forces a software change, not all of the software has to be changed. Software tasks are more or less isolated from one another, and have a very simple and precisely defined interface with one another via the operating system. One guideline used in MLT-2 is to partition the software in such a way that it maps onto the system hardware as much as possible. This mapping tends to make the software system change tolerant. Another guideline followed is to buffer the application tasks from the hardware drivers as much as possible. Again, a greater degree of change tolerance is achieved.

4.1 Input/Output structure

A unified I/O structure is used for all MLT-2 microcomputer circuits. The main idea is to buffer the user tasks from the hardware drivers (usually interrupt-level software functions), and to standardize the task/driver interface. The constraint imposed is that a task never communicates directly with a hardware driver. Instead, the task makes a function call, and passes some necessary parameters (the address of a message buffer, a byte count, and the address of a result flag). The function called is referred to as the interface function. A RAM control block for the I/O hardware is defined and known only to the interface function and to the driver software. The interface function does only two things. First, it fills the control block with the user parameters. Second, it performs one action to start the I/O. The I/O operation is

completed by the driver at interrupt level. The only connection between the driver and the interface function is the common control block. The only connection between the driver and the user task is a semaphore that is set by the driver when its activity is completed. The user task waits for the occurrence of tbe semaphore, after whicb it is scheduled to execute by m8os. Since the only interaction between the driver and the operating system is the setting of the user semapbore, an extremely fast and error-free interface is achieved between task, driver, and operating system.

4.2 Task structure

User tasks are defined so that the software maps onto the system hardware as mucb as possible in order to minimize the impact on the software of changes in the hardware or in the hardware drivers. Hence, each I/O facility of sufficient complexity has its driver and interface function, and also a unique task that controls that I/O facility. Any other task that needs I/O from (to) that device merely receives or sends a message buffer via m8os to the controlling task. The interface between any task and the I/O is now very simple. Since only one task controls the I/O, there is no confusion about when the device is ready for the next operation, and user tasks do not have access to global variables that describe the I/O function. Potential software errors are thereby avoided, and an application task can change and not complicate or even cause changes to the drivers.

Figure 5 shows the software architecture of the DCN third-tier function, and is representative of the architecture in the remaining DGN circuits. The large ovals indicate an application task, whereas the small ovals indicate an interface function. Because of the simplicity of the DGN operation, only three types of tasks are required. One controls the GPIB interface. Since a third-tier circuit contains four GPIBs (Fig. 4), four large ovals of this type are indicated. Although tbere are four distinct GPIB tasks, all execute the same program text. Similarly, there are eight BX.25 tasks shown in Fig. 5 to control the eight LTS data links. A message received over a GPIB is simply sent to the appropriate BX.25 task via the m8os message passing facility.

Each MLT-2 microcomputer contains an administrative task (ADMIN in Fig. 5). This module performs all nonoperational functions required in the local environment. For example, ADMIN controls sanity and diagnostic functions and the reporting of trouble count and usage statistics. The MLT-2 message header contains the message source and destination circuit types and task identifiers. Hence, ADMIN tasks in different microcomputer modules can communicate to synchronize actions across circuit-board boundaries. Lack of space prohibits a more detailed discussion of MLT-2 microcomputer sanity and diagnostic strategy and software.

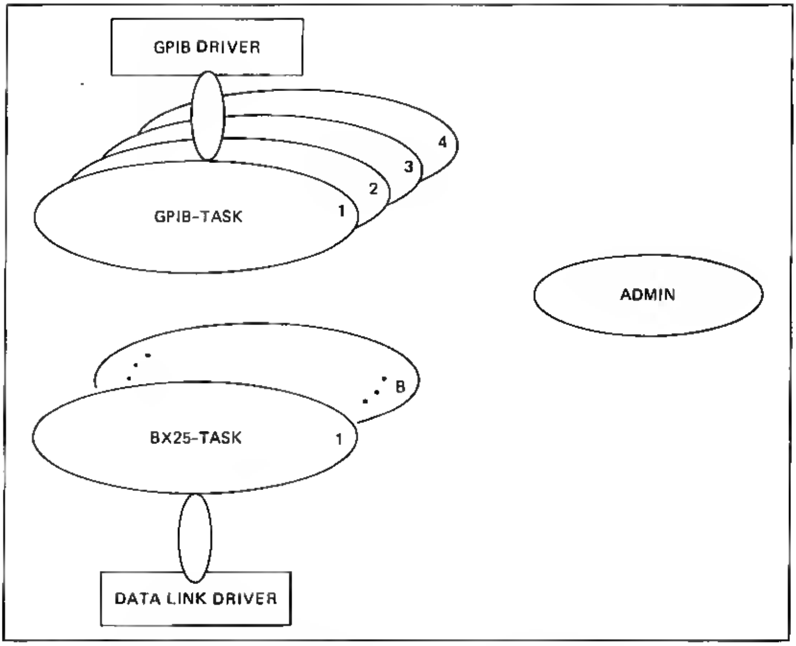


Fig. 5—Data Communication Network software structure.

Other MLT-2 microcomputer modules have software architectures that are similar to that shown in Fig. 5. For example, the port controller contains a GPIB task to control its communication interface. It also contains one PORT task for each port in the LTS (i.e., 16 tasks). Each PORT task can control all activities on a particular port in the system. If the GPIB task receives an access request, it passes the message to a PORT task, and access is controlled from this software function. All 16 PORT tasks execute the same program text.

The reader should note the similarity that exists in all MLT-2 microcomputer software environments. This similarity is of course intentional, and provides several benefits. Different MLT-2 microcomputer circuits have the same types of I/O interfaces. The structure allows us to include a single software module in all environments where it can be used. Similar structures make it easier to understand and cope with the large amount of software required to produce the MLT-2 functions.

V. CONCLUSIONS

The second-generation MLT is a microprocessor-based distributed processing system that performs the loop testing function in the ARSB. The main architectural technique used in the design of MLT-2 is to

distribute the access, test, and communications functions as closely as possible to the point of testing. New architectural components include the LTS and the DCN. The LTS can appear in each wire center served by the system, and is composed of a collection of loosely coupled microprocessors. One of these is a sophisticated general-purpose testing instrument, the PMU. The architecture and operation of the LTS are discussed in some detail. The DCN allows any LMOS/MLT FE to exchange data with any LTS. The DCN is itself a collection of loosely coupled microprocessors, and is constructed from five basic circuits. The modularity and fault-tolerance of the DCN are discussed.

The architecture of the MLT-2 microcomputer software is described. The multitasking environment is explained, and techniques for dealing with this environment are presented. Major software design goals are to make the system tolerant to change, easy to understand, and easy to maintain. These goals are achieved by standardizing I/O interfaces, partitioning software so that it maps onto the system hardware and/or the problem being solved, and making use of functional commonality across the different MLT-2 microcomputer software environments.

The use of a distributed testing architecture provides many advantages to the ARSB FE complex, including the ability to support testing in a very large number of wire centers. Installations of MLT-2 can have many hundreds of microprocessors involved in the testing function for the totality of loops served. The functional decomposition techniques used in the design of MLT-2 help deal with the complexity that accompanies this large distributed processing system.

VI. ACKNOWLEDGMENTS

Many people have made significant contributions to the MLT-2 development. I gratefully acknowledge the efforts of the members of the Maintenance Systems Engineering Department and the Mechanized Loop Testing Department.

REFERENCES

1. F. J. Uhrhane, "Automated Repair Service Bureau: Mechanized Loop Testing Strategies and Techniques," B.S.T.J., this issue.
2. O. B. Dale, T. W. Robinson, and E. J. Theriot, "Automated Repair Service Bureau: Mechanized Loop Testing Design," B.S.T.J., this issue.
3. E. A. Overstreet, "Automated Repair Service Bureau: Economic Evaluation," B.S.T.J., this issue.
4. C. Weitzman, *Distributed Micro/Minicomputer Systems,* Englewood Cliffs, New Jersey: Prentice-Hall, 1980.
5. S. Muftic and N. Husovic, "On Functionally Distributed Computing Systems," IEEE Symp. on Trends and Applications: Distributed Processing, 1978.
6. *Terminals and Communications Handbook,* Digital Equipment Corporation, pp. 272–5, 1979.
7. R. L. Martin, "Automated Repair Service Bureau: The System Architecture," B.S.T.J., this issue.
8. C. M. Franklin and J. F. Vogler, "Automated Repair Service Bureau: Data Base System," B.S.T.J., this issue.

9. G. H. Leonard and J. E. Zielinski, "Automated Repair Service Bureau: Human Performance Design Techniques," B.S.T.J., this issue.
10. T. Holub, "Bell Labs Announces Powerful New Microprocessor for Wide Range of Bell System Applications," Bell Labs News, February 17, 1977, Murray Hill, N.J.: Bell Telephone Laboratories.
11. *IEEE Standard Digital Interface for Programmable Instrumentation*, New York: Institute Electrical & Electronics Engineers, 1978.
12. *Operations Systems Network Protocol Specification: BX.25*, Issue 2, Bell Telephone Laboratories, March, 1980.
13. J. S. Thompson and J. R. Boddie, "An LSI Digital Signal Processor," Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing, Denver, Colorado, April, 1980.